

# Class-Level Variables in Alice



By Jenna Hayes  
Under the direction of Professor Rodger  
Duke University, July 2008

# Instructions for the Game

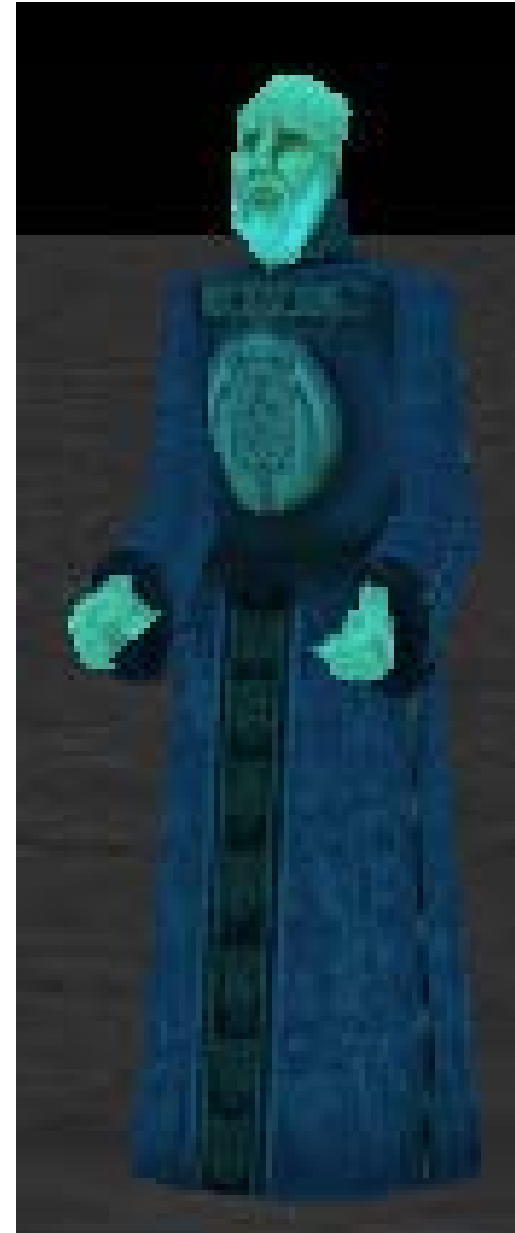
Download the starting world for this tutorial. It includes a game. These are the instructions:



- The point of the game is to collect the 6 lighted cubes that will hide themselves at the beginning of the game.
- You can press **F** to make your character face towards the nearest cube.
- You can press **T** to teleport to a different location.
- Your task is to limit the wizard's power so you can't just press **F** 5 times to win. You also need to make it so that when the wizard collects all 6 cubes, the game winning animation shows up.

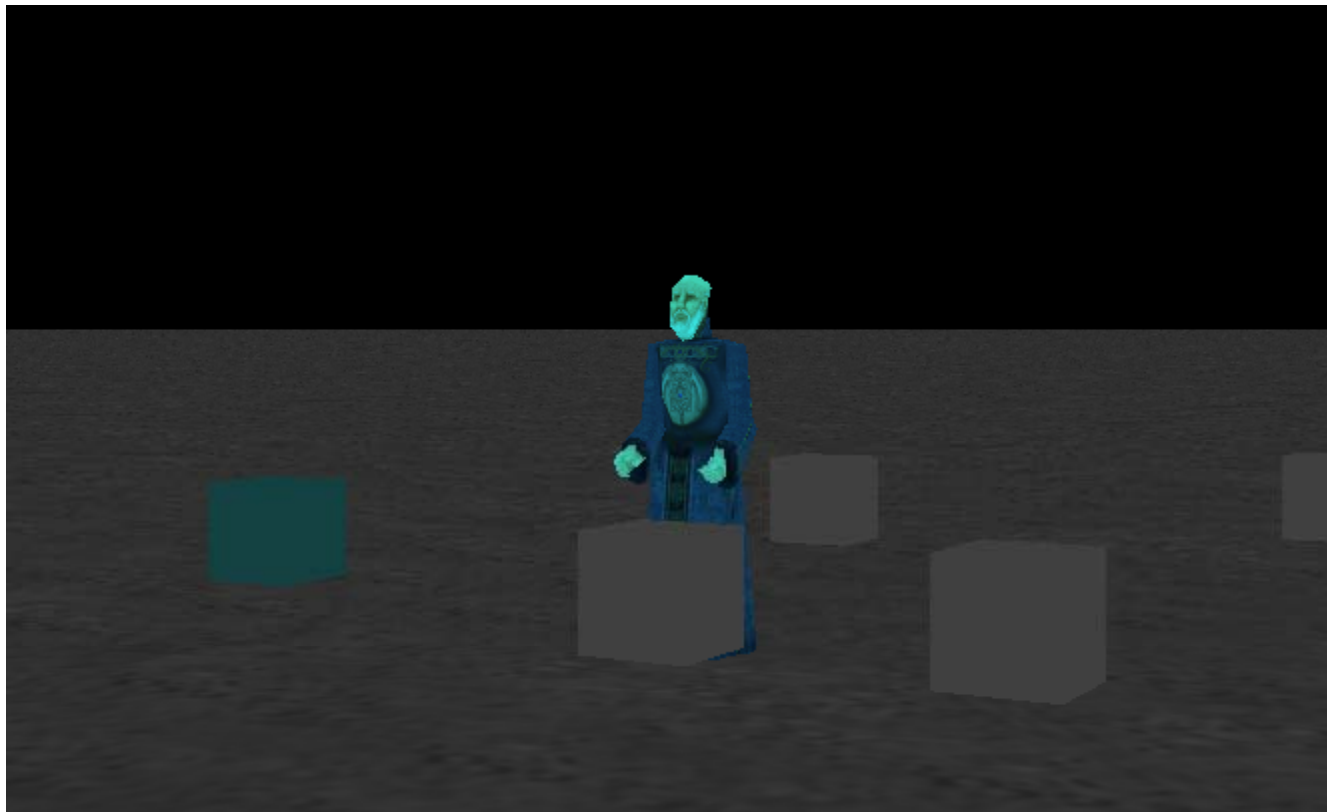
# Using Variables

- A **variable** is a space where information that changes can be stored, such as a number that needs to be added to or subtracted from as a game is played.
- A **class-level variable** is information that changes and is tied to a specific object, as opposed to just general information.



# Testing Out the World

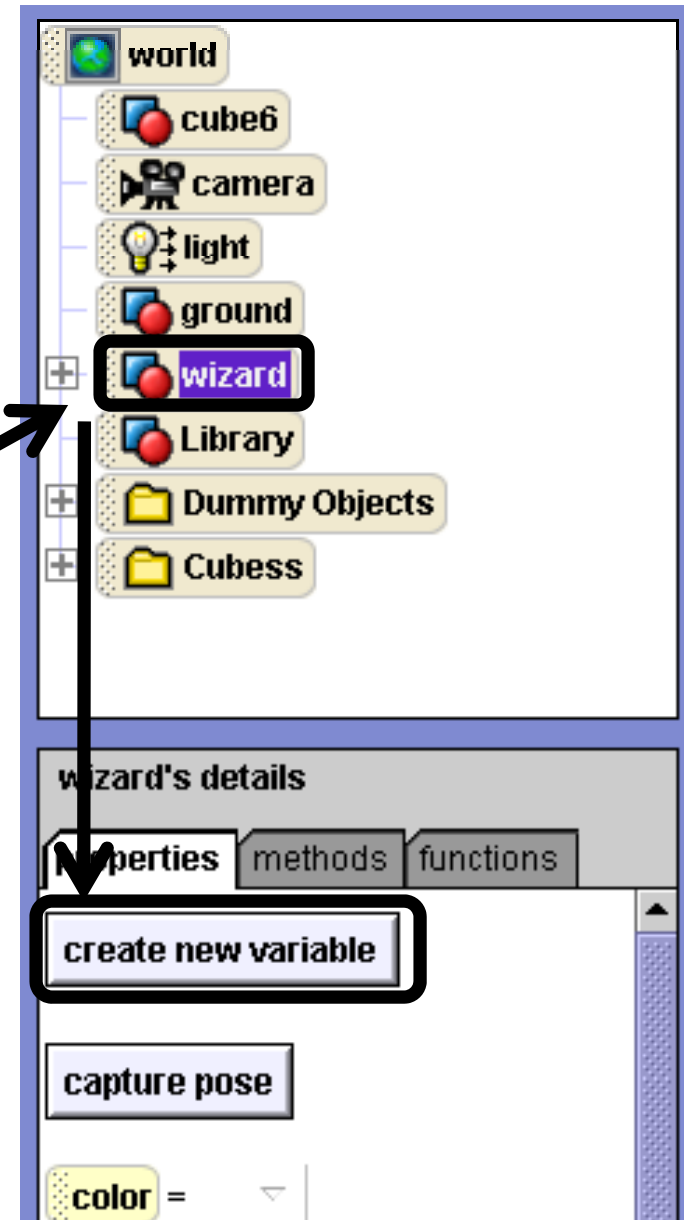
Try playing the world to see what happens and familiarize yourself with how the game looks. The opening animation should play in which the lighted cubes fly off to different places in the world.



# Creating the Variables

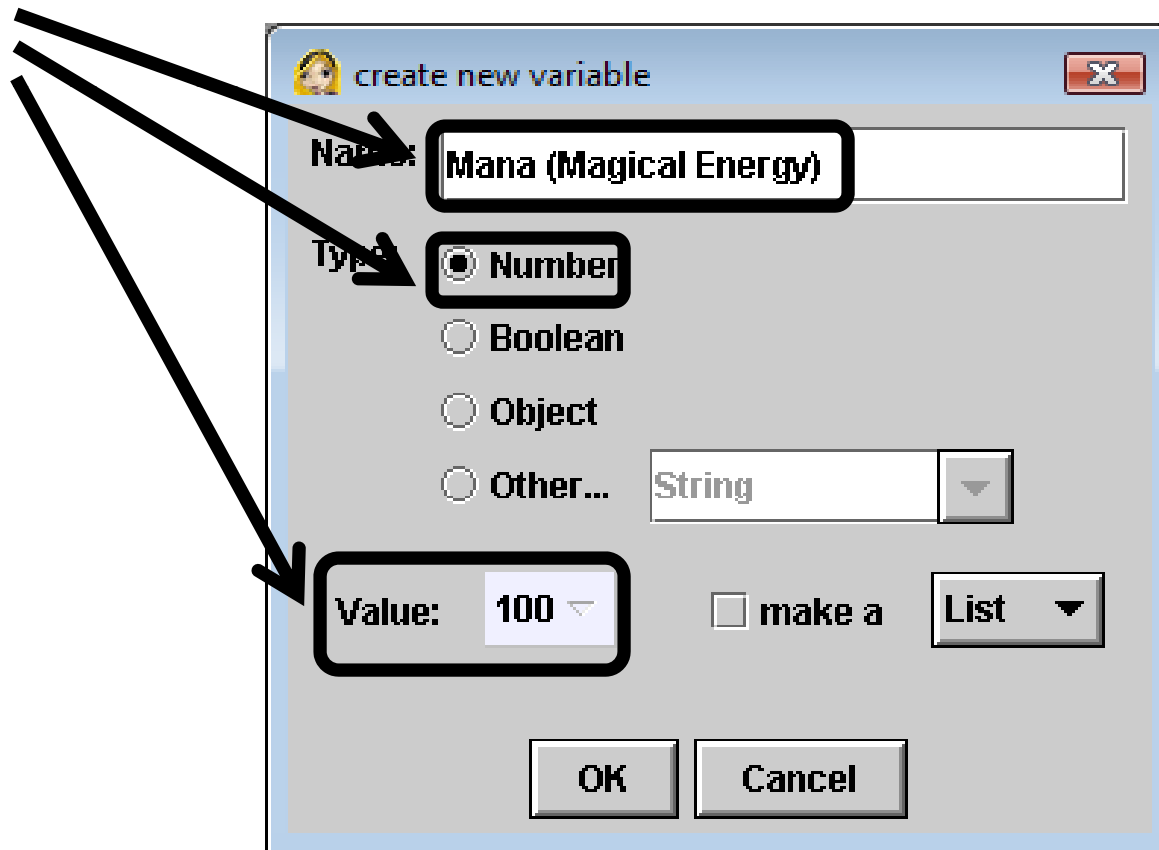
-We are going to give the wizard **mana**, which in typical role-playing video games are points that a character can use to perform magic.

-Click on the **wizard** in the object tree, and then on the **properties** pane click **create new variable**.



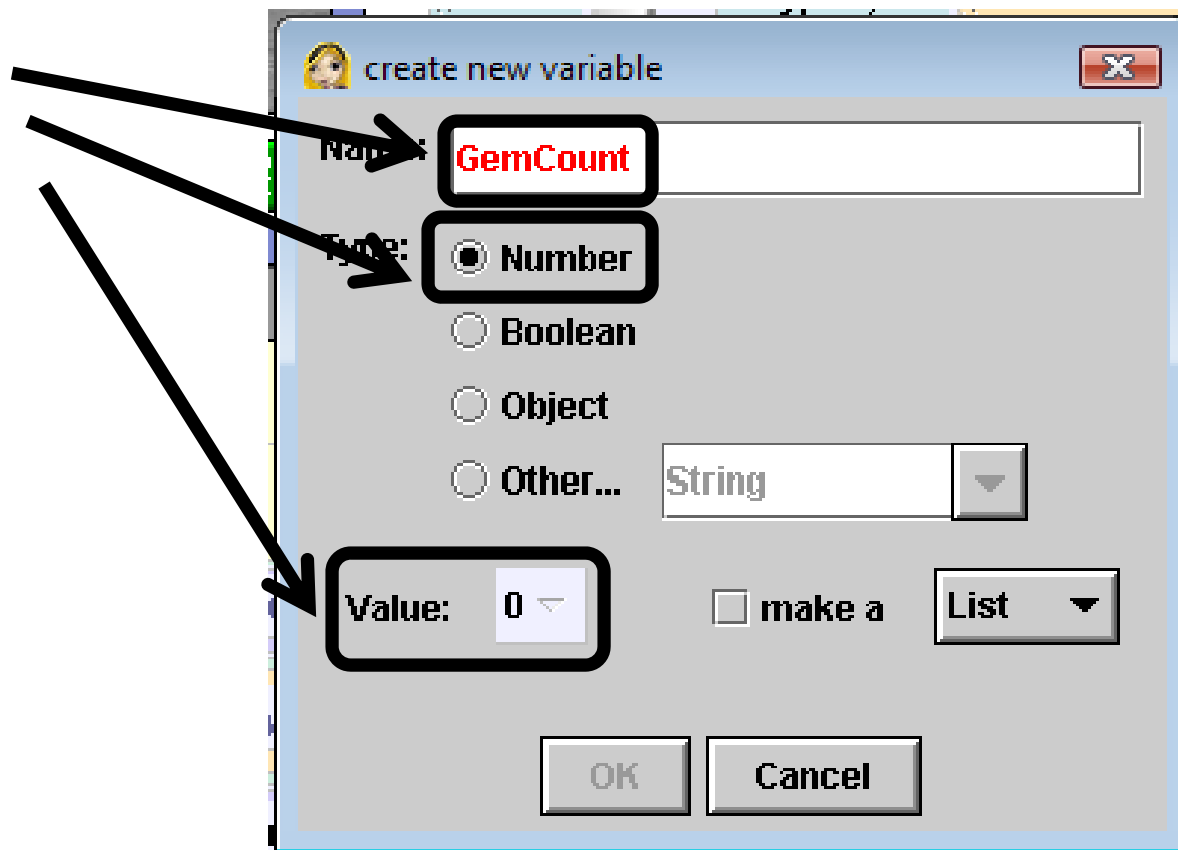
# Creating the Variables

Name your variable **Mana (Magical Energy)**.  
Make sure that it is a **number** variable, and  
set its initial value to **100**.



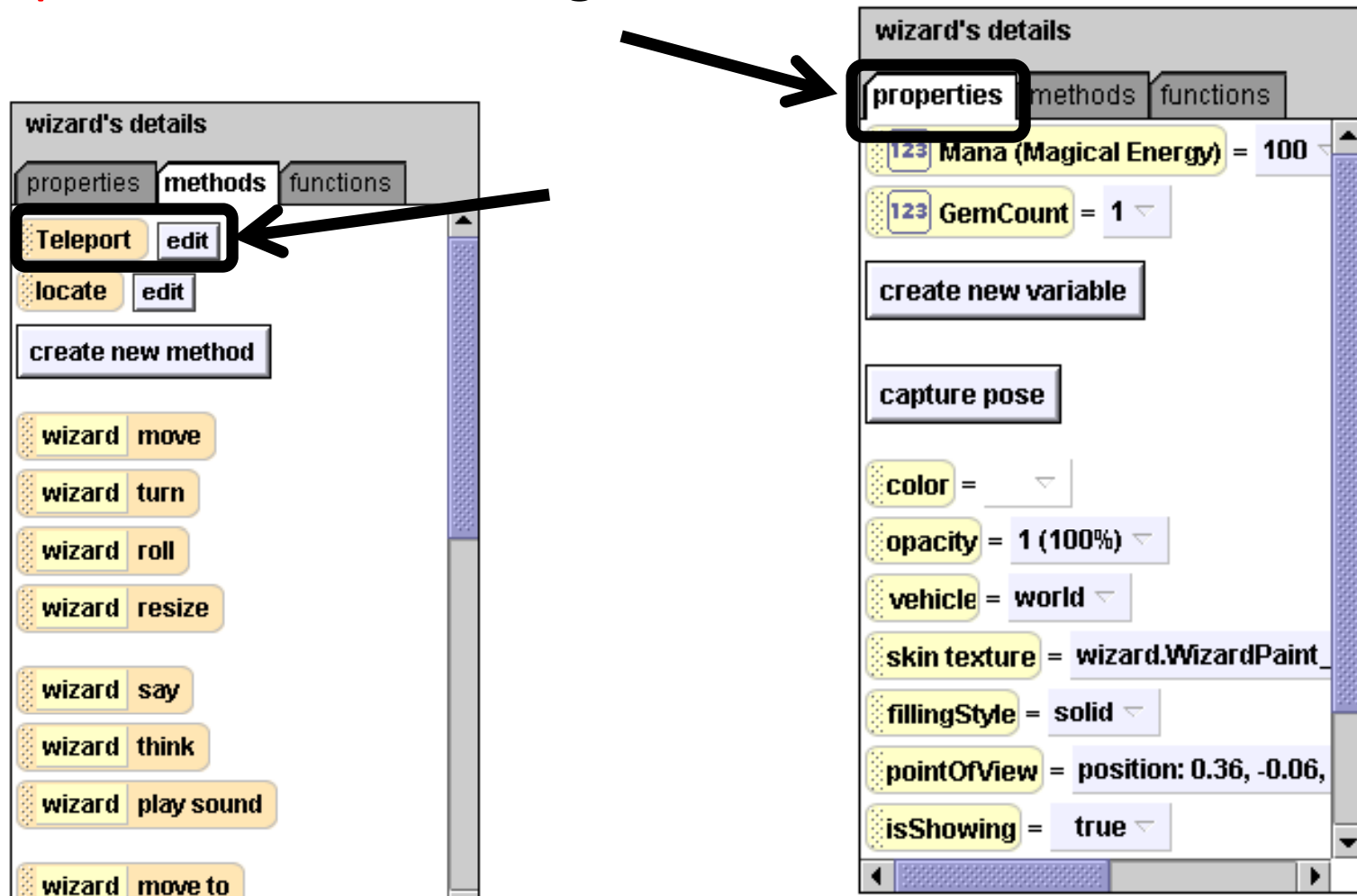
# Creating the Variables

Now create another variable to store the number of gems that the wizard has collected so far. This will help so we can create a condition so we know when the game has been won. Call it **GemCount** and set its starting value at **0**.



# Setting Up the Mana

-Now we want to make it so that when the wizard does magic (either **teleport** or **locate**) his **mana** goes down. Click on the **wizard** in the object tree. Go to **methods** and click **edit** next to the **Teleport** method. Then go to the wizard's **properties** pane.





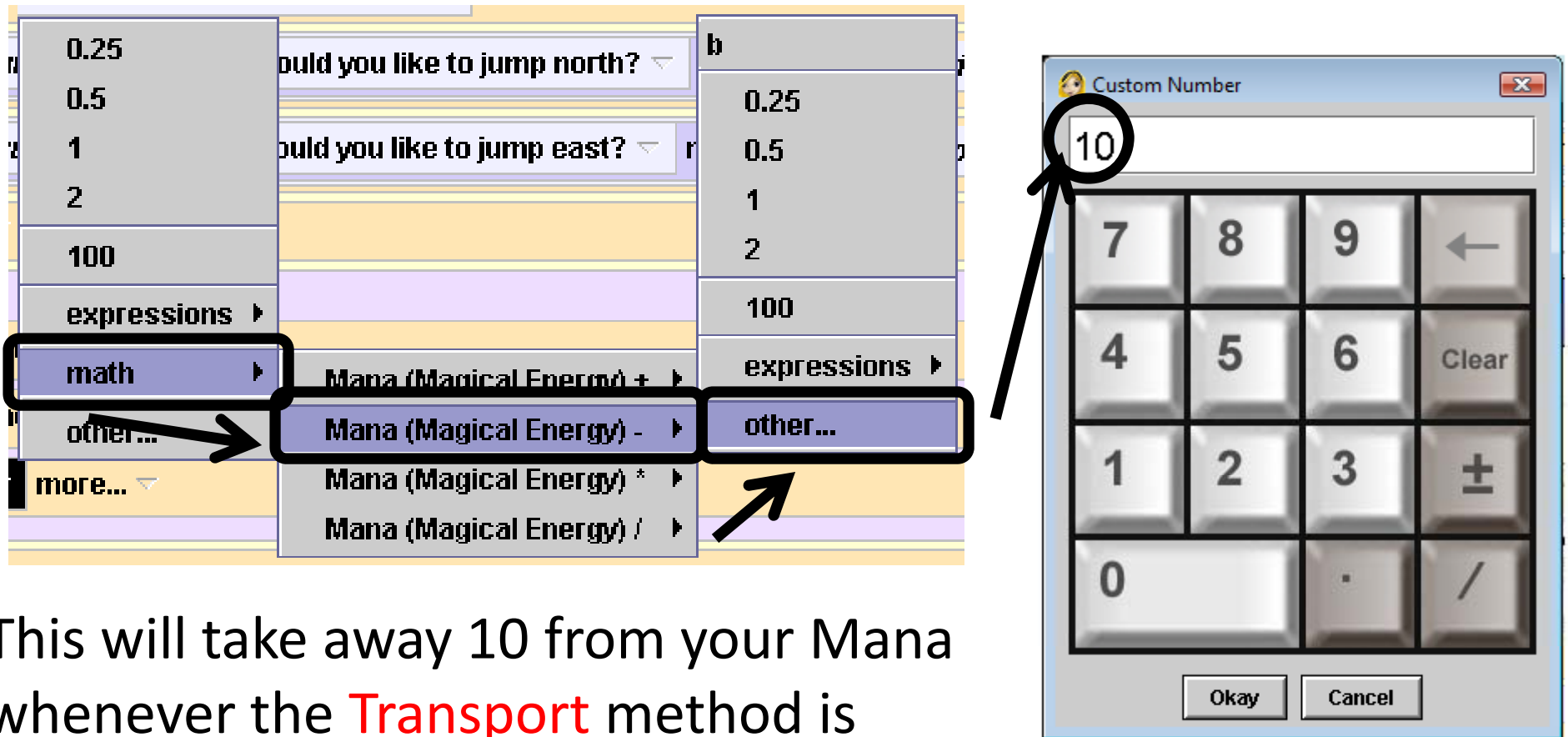
# Setting Up the Mana

Now drag-and-drop the **Mana (Magical Energy)** variable to the top of the **Teleport** method. Then select **set value**, **expressions** and **wizard.Mana(Magical Energy)**. Your code will look like this when you're done:



# Setting Up the Mana

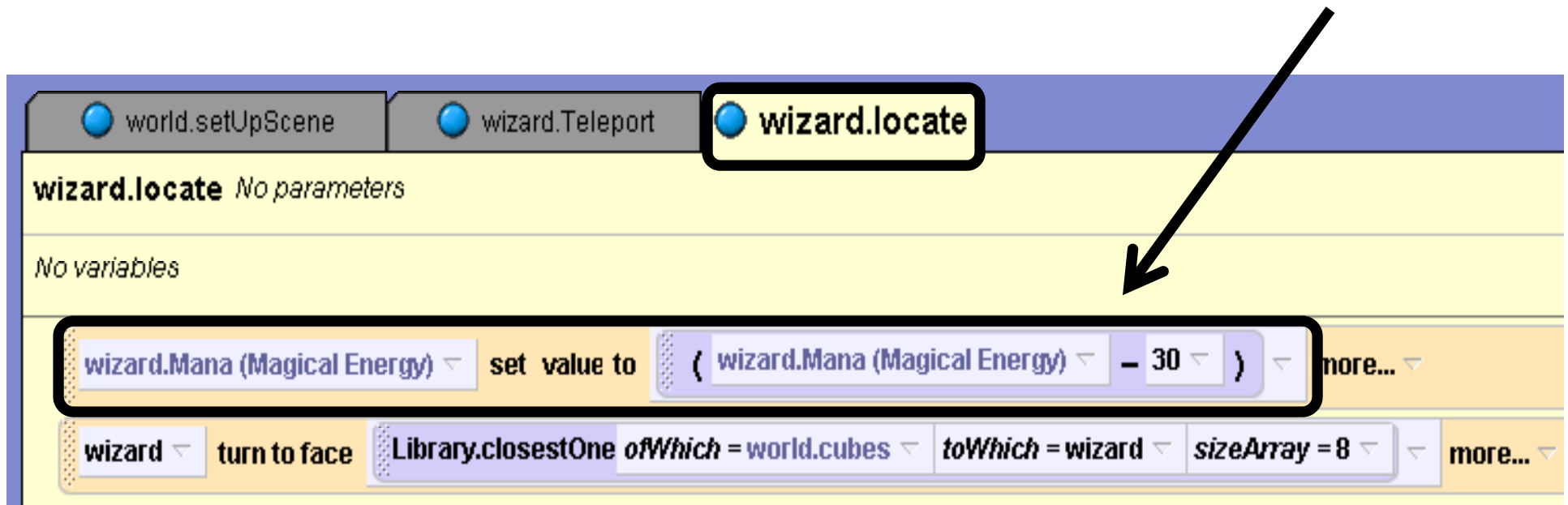
Click the down arrow next to the second **wizard.Mana(Magical Energy)** and then select **math**, and then **Mana(Magical Energy)-** and then **other...**. On the calculator that pops up, enter in **10**, and then click **Okay**.



This will take away 10 from your Mana whenever the **Transport** method is called.

# Setting Up the Mana

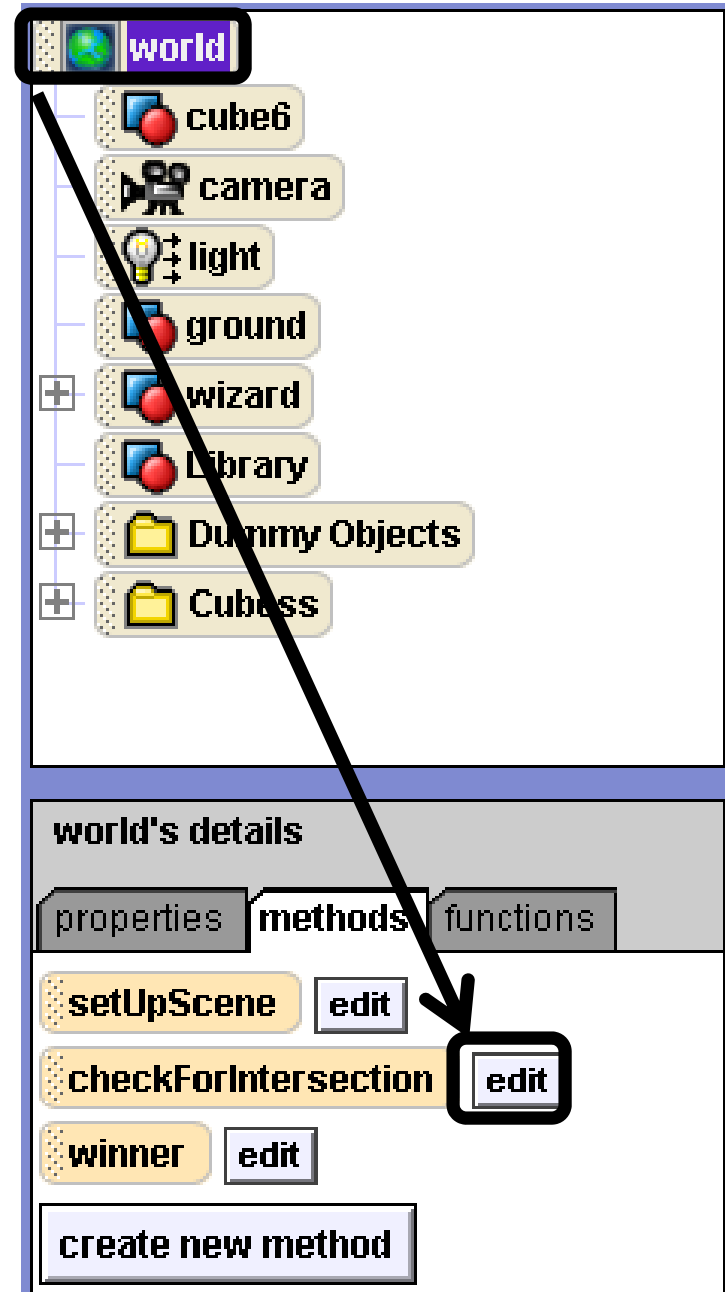
Now repeat the same process for the wizard's **locate** method. This time, instead of subtracting **10** from the mana, subtract **30**. Your **locate** method will look like this when you're done:



This will subtract 30 from your mana whenever you press **F** to use the **locate** method.

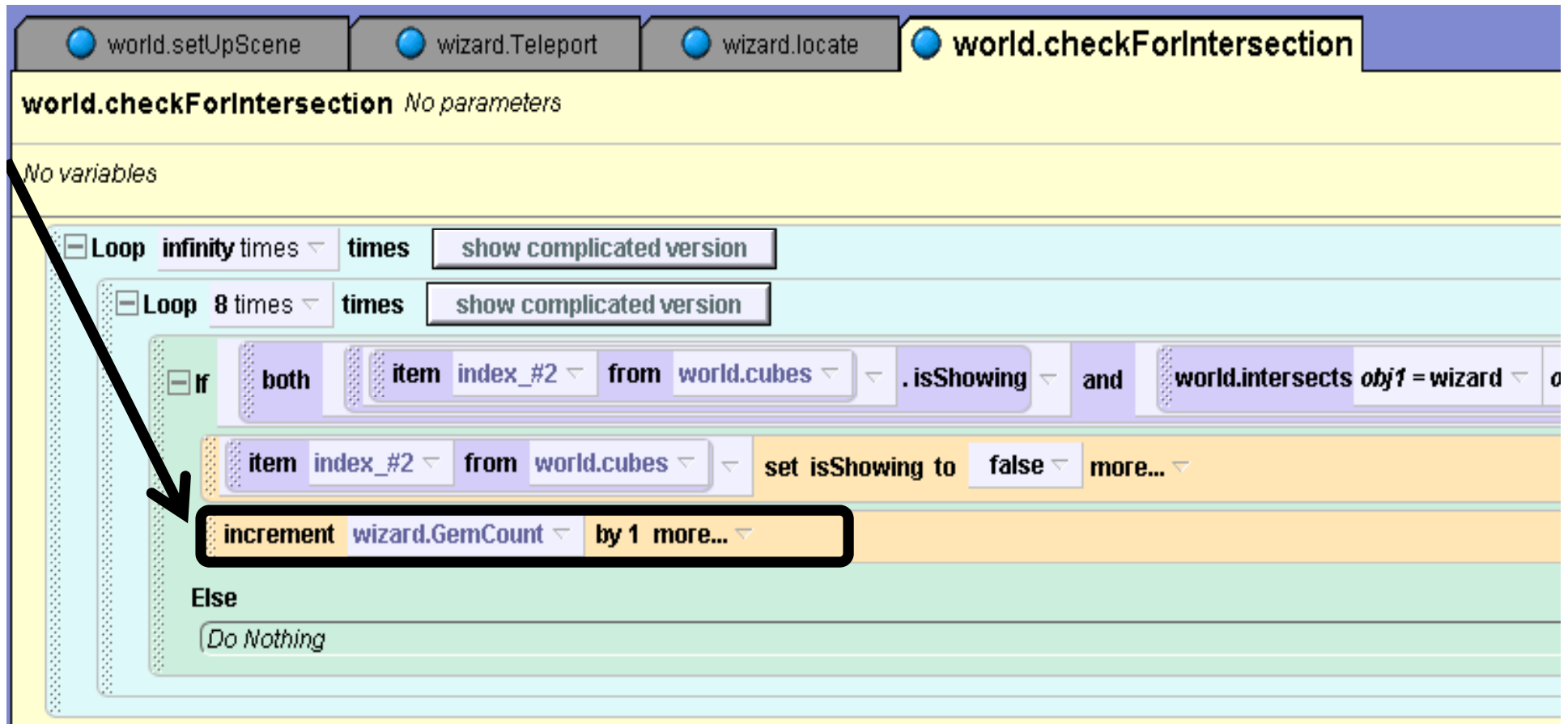
# Setting Up the GemCount

Now we want the **GemCount** to increase whenever a gem is found. Click on **world** in the object tree, and then go to the **methods** pane. Click on **edit** next to the **checkForIntersection** method.



# Setting Up the GemCount

Now click on **wizard** in the object tree and go to the **properties** pane. Drag the **GemCount** variable into the **If Else** statement and drop it there. Then select **Increment wizard.GemCount by 1**. Your **checkForIntersection** method will look like this when you're done:



# Creating Restrictions

Now we need to restrict the wizard so that he can't keep using negative mana into infinity. We need to make it so that when he runs out, he can no longer do **Teleport** or **Locate**. Go back to the **Teleport** method, and drag an **If Else** statement to the top of it, selecting **true**.

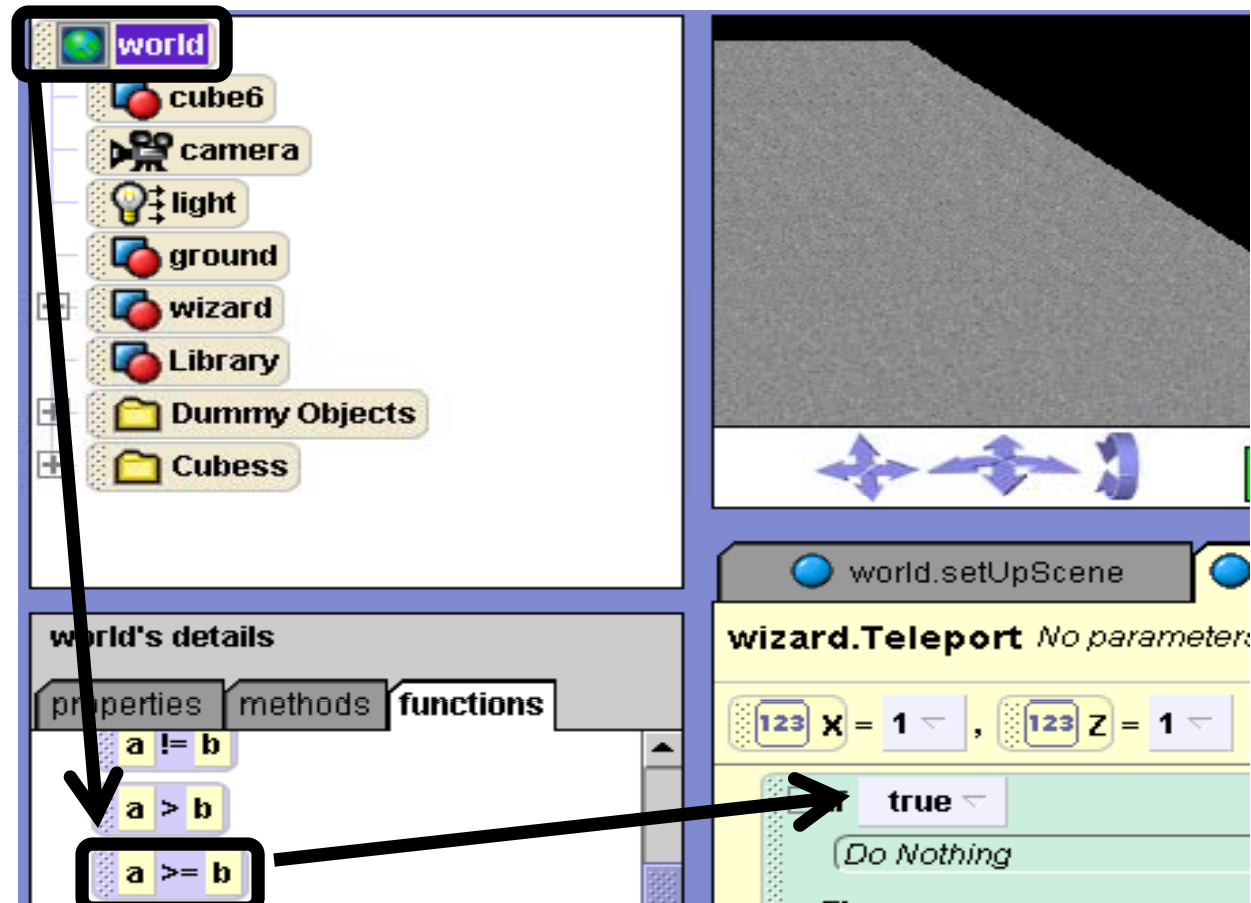
The image shows the Scratch script editor for the **wizard.Teleport** method. The script is as follows:

- If true** (selected)
  - Do Nothing
- Else**
  - Do Nothing
- wizard Mana (Magical Energy) set value to ( wizard Mana (Magical Energy) - 10 ) more...
- X set value to ask user for a number question = How far would you like to jump north? more... duration = 0 seconds more...
- Z set value to ask user for a number question = How far would you like to jump east? more... duration = 0 seconds more...
- camera set vehicle to world more...
- Do together**
  - wizard set emissiveColor to [black] more...

An arrow points from the **If/Else** block in the bottom palette to the **If true** block in the script. The **If/Else** block is highlighted in the bottom palette.

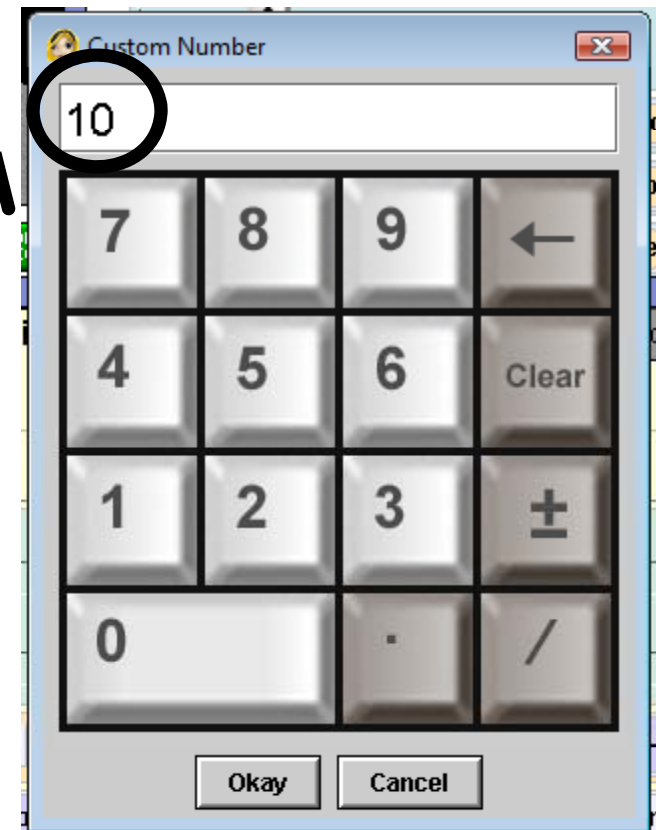
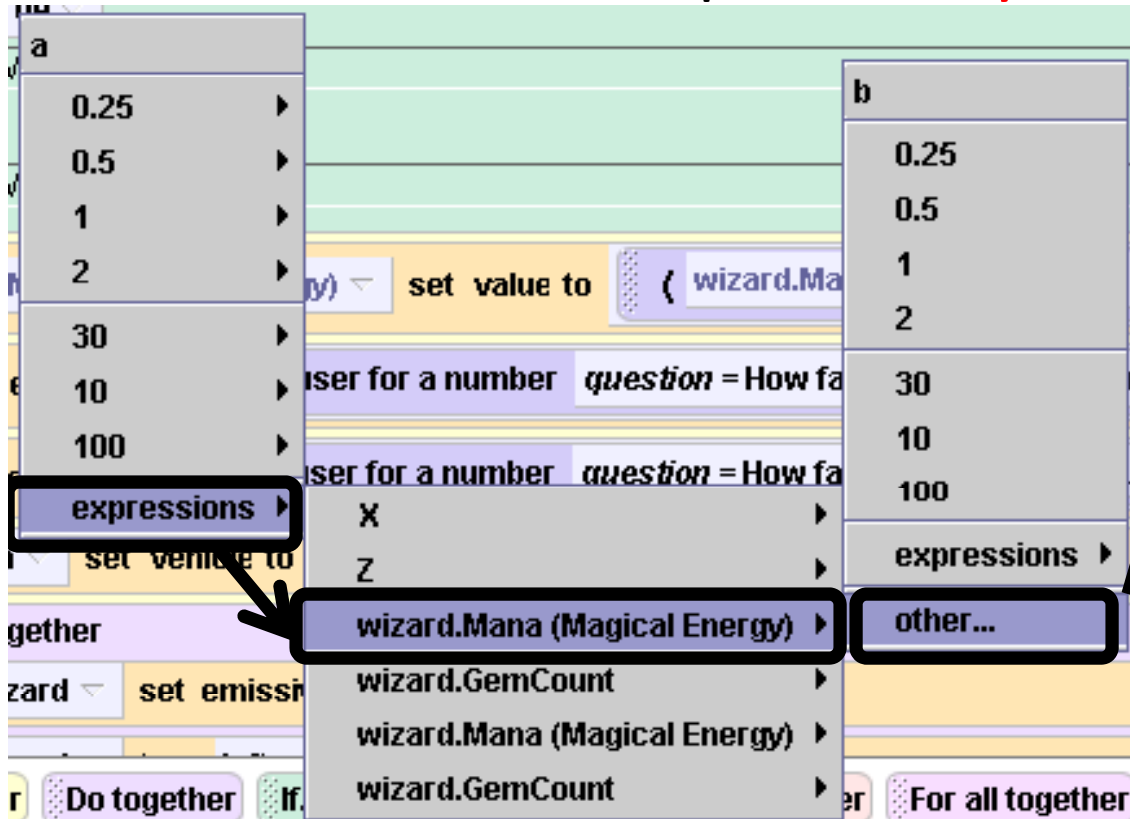
# Creating Restrictions

Now, we only want him to teleport if his Mana is greater than or equal to **10**, because that's how much Mana it takes to teleport. For this, we need a **>=** sign. Click on **world** in the object tree and then go to the **functions** pane. Find **a >= b** and drop it over the **true** on your If statement.

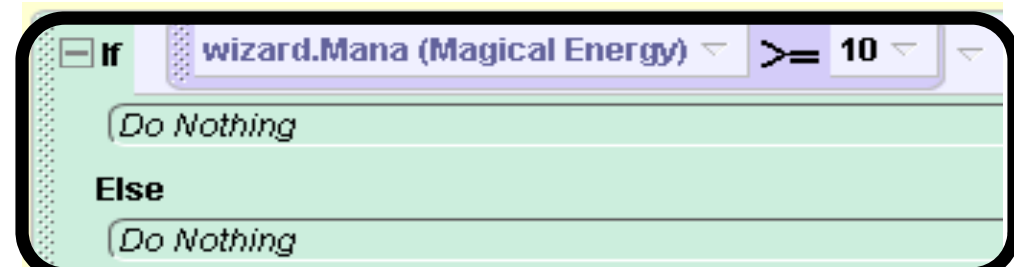


# Creating Restrictions

On the menu that pops up, select **expressions**, then **wizard.Mana(Magical Energy)**, then **other...**. Then type in **10** on the calculator, and press **Okay**.



Your finished **If Else**  
will look like this: →





# Creating Restrictions

Now, we only want **Teleport** to work if the mana is greater than 10. This means we have to drag and drop *all* of the code in the rest of the method into the top of the **If Else** statement. You will have to drag each piece one at a time, making sure to keep them in the right order. See the next slide to see what your **Teleport** method will look like.

The image shows a Scratch script editor with four tabs: **world.setUpScene**, **wizard.Teleport** (selected), **wizard.locate**, and **world.checkForIntersection**. The **wizard.Teleport** script has the following components:

- Script title: **wizard.Teleport** No parameters
- Buttons: **create new parameter** and **create new variable**
- Code blocks:
  - 123 X = 1** and **123 Z = 1** (variable blocks)
  - If** statement:
    - Condition: **wizard.Mana (Magical Energy) >= 10**
    - Do Nothing** (true branch)
    - Else** (false branch)
      - Do Nothing** (initial block)
      - wizard.Mana (Magical Energy) set value to ( wizard.Mana (Magical Energy) - 10 ) more...** (highlighted in a red box)
      - X set value to ask user for a number question = How far would you like to jump north? more... duration = 0 seconds mor** (highlighted in a red box)
      - Z set value to ask user for a number question = How far would you like to jump east? more... duration = 0 seconds more** (highlighted in a red box)

A black arrow points to the **Else** block, indicating where the code from the rest of the method should be moved.

**If** wizard.Mana (Magical Energy)  $\geq$  10



wizard.Mana (Magical Energy) set value to ( wizard.Mana (Magical Energy) - 10 ) more...

X set value to ask user for a number question = How far would you like to jump north? more... duration = 0 seconds more...

Z set value to ask user for a number question = How far would you like to jump east? more... duration = 0 seconds more...

camera set vehicle to world more...

**Do together**

- wizard set emissiveColor to  more...
- wizard turn left 5 revolutions more...
- world set ambientLightColor to  more...



camera set vehicle to wizard more...

Library.vectorMoveRel obj = wizard x = X y = 0 z = Z duration = 1 style = 1

light set brightness to 5 more...

Wait 2 seconds

**Do together**

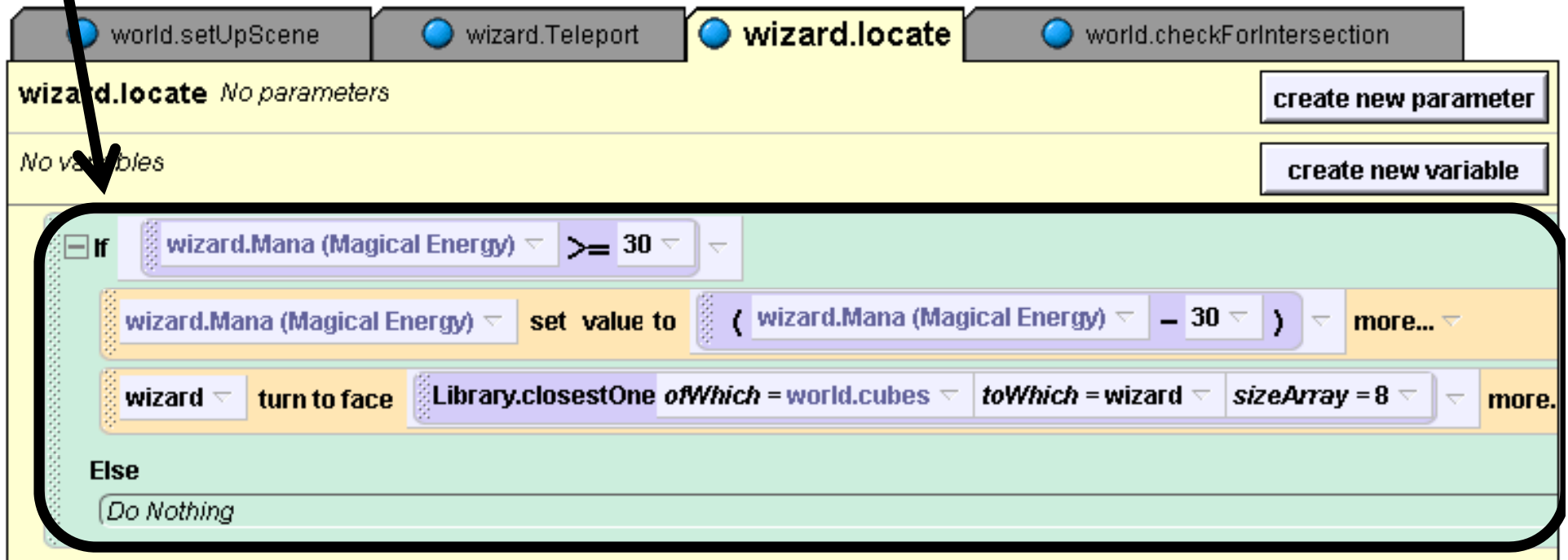
- light set brightness to 0 more...
- wizard set emissiveColor to  more...
- world set ambientLightColor to  more...

**Else**

Do Nothing

# Creating Restrictions

Now do the same steps with your **Locate** method. In an **If Else** statement, make the statement **If wizard.Mana(Magical Energy)>=30**, and drag all of the code in the method into it. It will look like this when you're done:



The image shows a Scratch script editor with four method tabs: `world.setUpScene`, `wizard.Teleport`, `wizard.locate` (selected), and `world.checkForIntersection`. The `wizard.locate` method is expanded, showing its parameters and variables. A black arrow points to the `wizard.locate` tab. The script contains an **If-Else** statement:

- If** `wizard.Mana (Magical Energy) >= 30`
  - `wizard.Mana (Magical Energy)` set value to `( wizard.Mana (Magical Energy) - 30 )`
  - `wizard` turn to face `Library.closestOne` ofWhich = `world.cubes` toWhich = `wizard` sizeArray = `8`
- Else**
  - `Do Nothing`

# Ending the Game

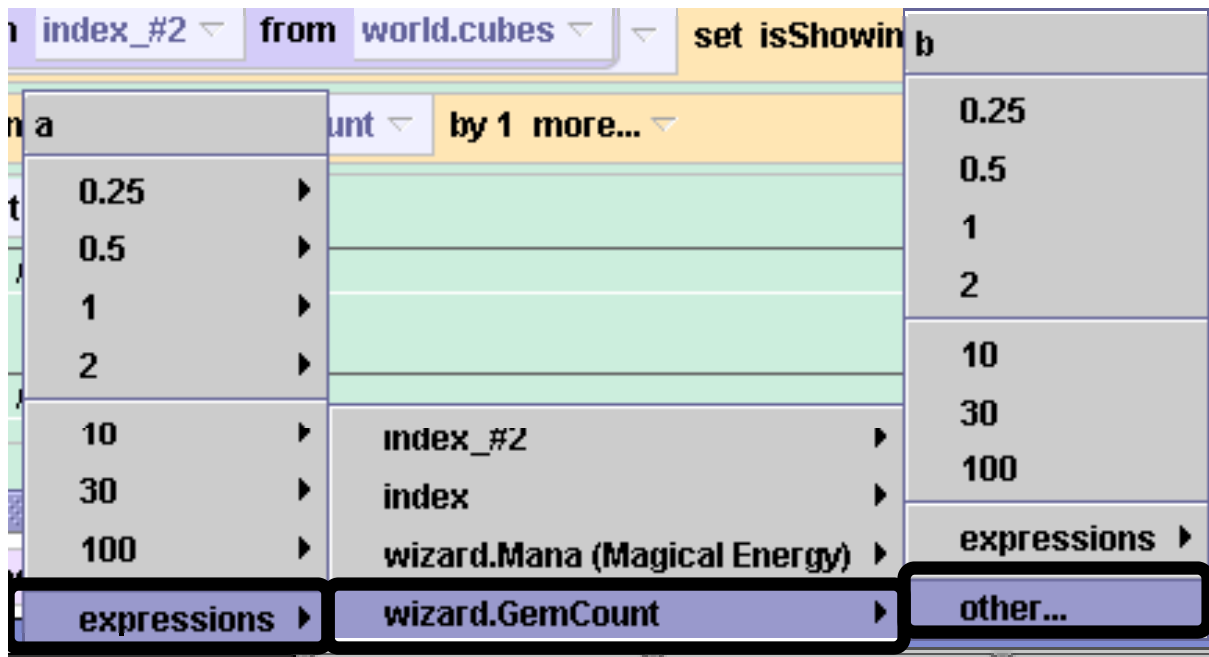
Now we need to make it so that when all 6 gems are collected, the winning sequence plays (located in **world.winner**). Open up **world.checkForIntersection**. Add an **If Else** statement right below the **increment** method.

The image shows a Scratch script editor with the following components:

- Script Area:**
  - Buttons at the top: `world.setUpScene`, `wizard.Teleport`, `wizard.locate`, and `world.checkForIntersection` (highlighted with a black box).
  - Script title: `world.checkForIntersection` No parameters
  - Variables: No variables
  - Script blocks:
    - Loop: infinity times (dropdown: times) show complicated version
    - Loop: 8 times (dropdown: times) show complicated version
    - If statement (dropdown: both) with conditions:
      - item index\_#2 from world.cubes .isShowing and world.intersects on
    - Block: item index\_#2 from world.cubes set isShowing to false more...
    - Block: increment wizard.GemCount by 1 more...
    - If/Else statement (highlighted with a black box and an arrow):
      - If true: Do Nothing
      - Else: Do Nothing
- Block Palette:**
  - Buttons: Do in order, Do together, If/Else (highlighted with a black box), Loop, While, For all in order, For all together, Wait, print, and a comment block.

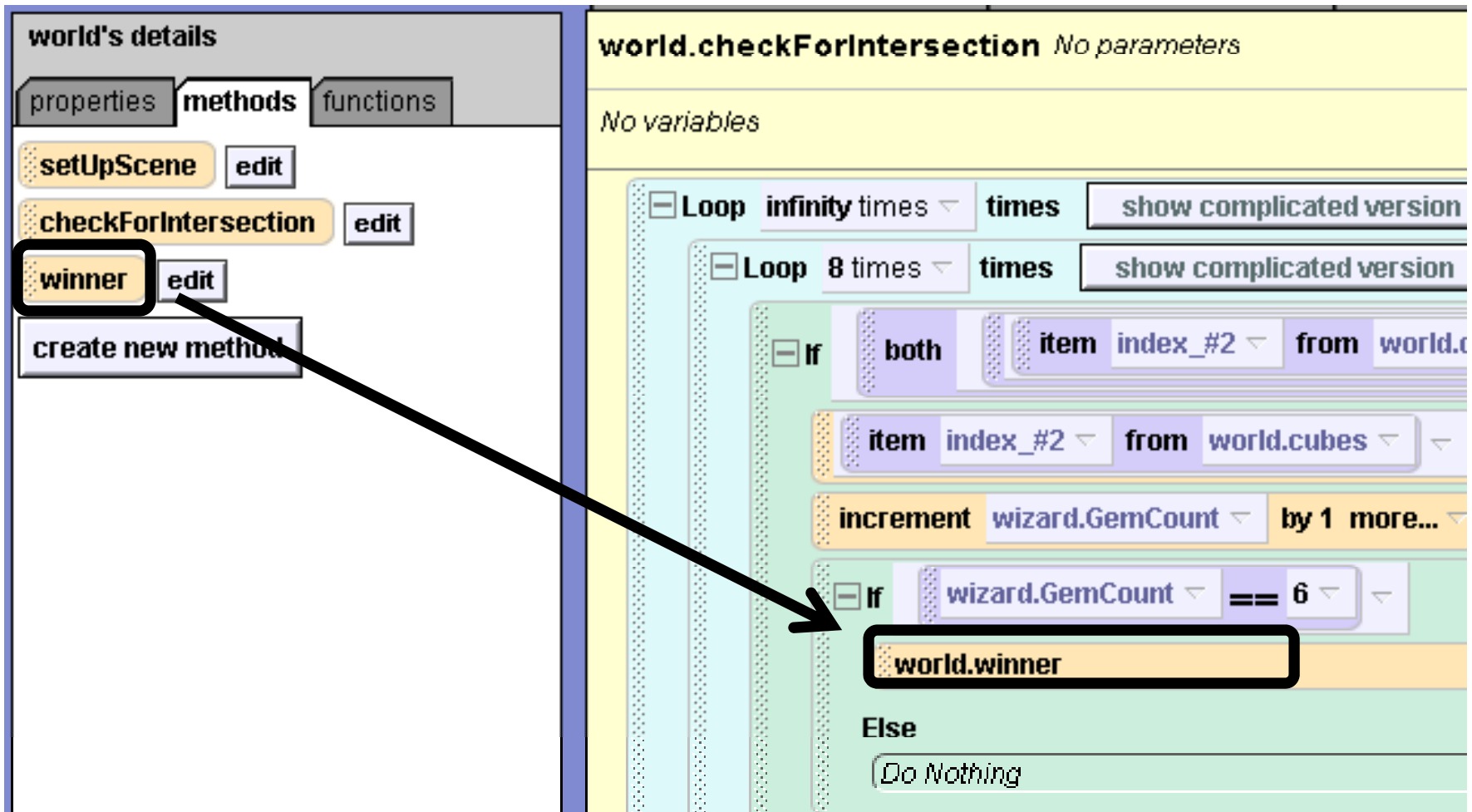
# Ending the Game

Now we need to complete the If statement. Click on **world** in the object tree and go to the **functions** pane. Find **a==b** and drag and drop it over the **true** in your If Statement. Select **expressions**, then **wizard.GemCount**, then **other...**. Type in **6** on the calculator and click **Okay**.



# Ending the Game

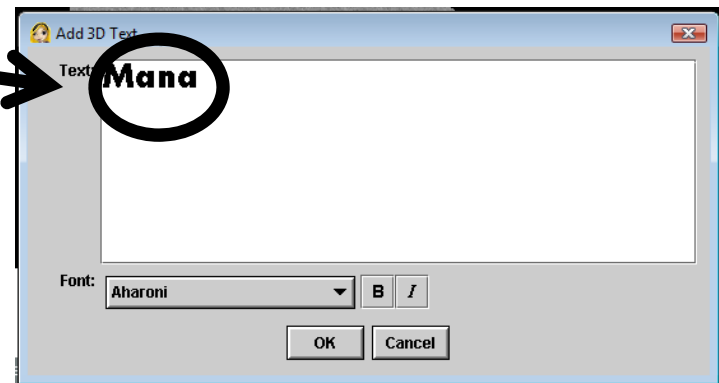
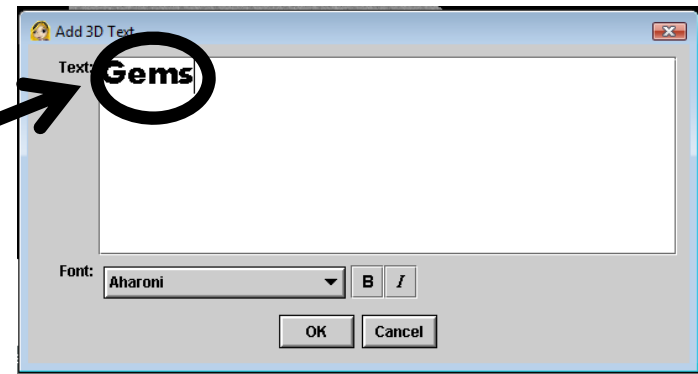
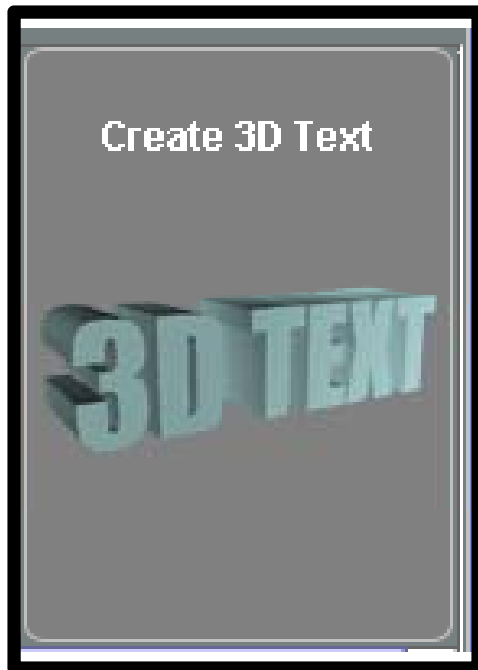
Now go into the world's **methods** pane, and drag and drop **winner** into your **If Else** statement.



Now your game should be functional!

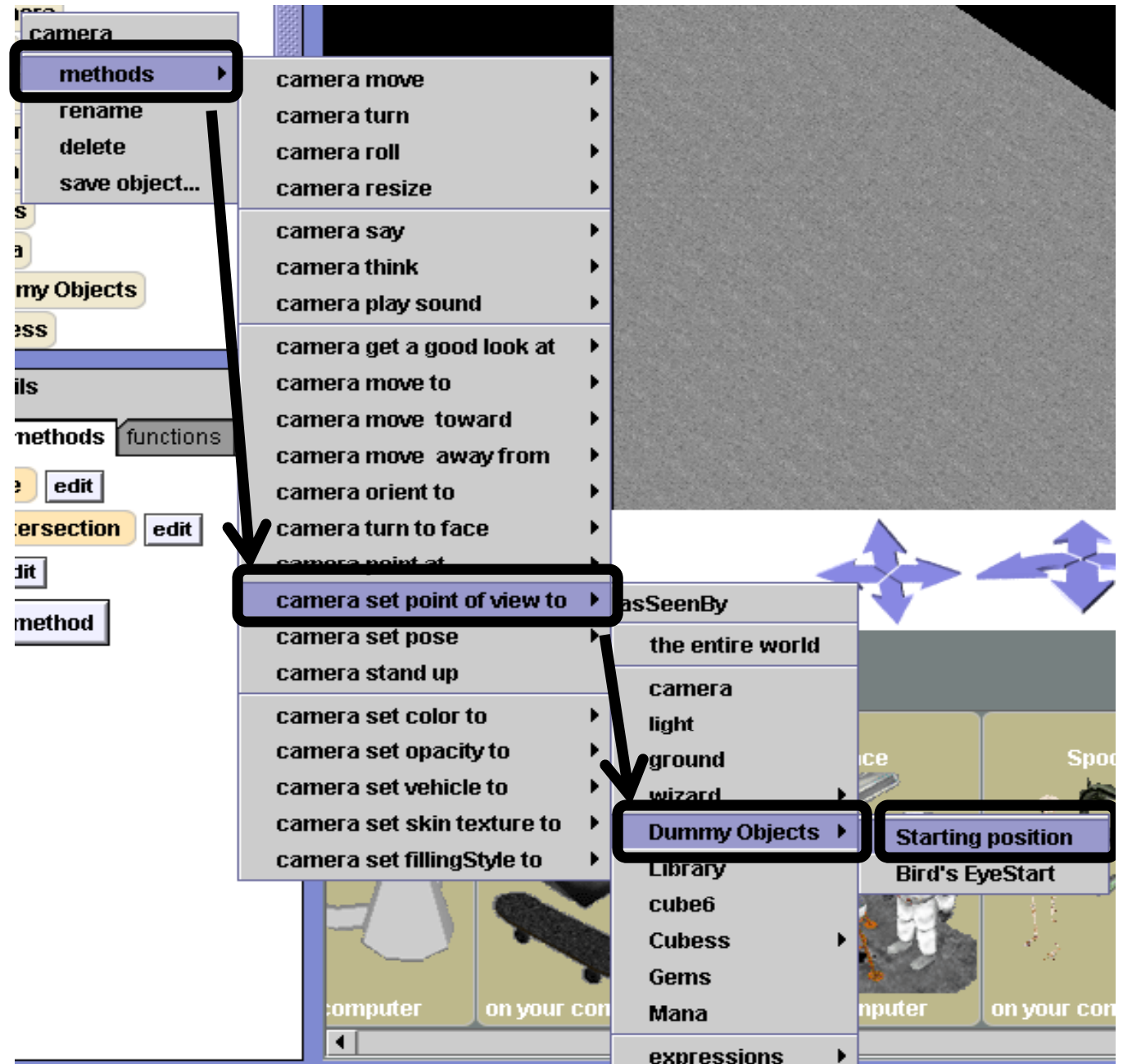
# Showing **Mana** and the **GemCount**

Now there's only one problem: we can't tell how much mana we have left, or how many gems we have collected, unless we keep track in our heads! We can fix this by adding **text objects** that display both of these numbers. Go to the **add Objects** screen and scroll to the end of the folders to find **3-D text object**. Add two of these, one called **Gems** and one called **Mana**.



# Positioning the Text Objects

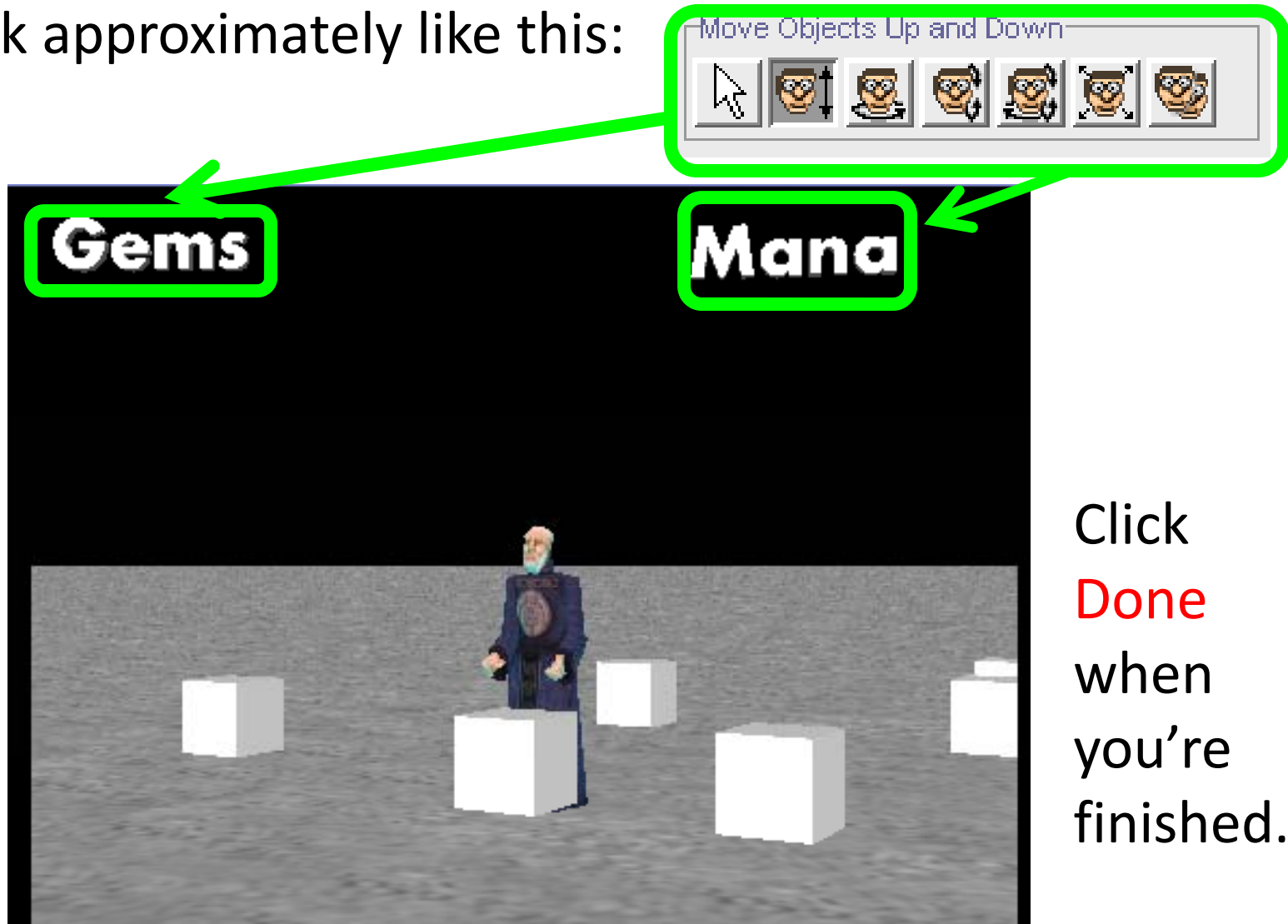
Now you need to move your camera position so you can see the text objects and the wizard. Right click on **camera** in your object tree, then select **methods**, then **camera set point of view to**, then **Dummy Objects**, then **Starting Position**.





# Positioning the Text Objects

Now use your object positioning buttons to push the text objects back from the camera, and then up in the sky, until they look approximately like this:

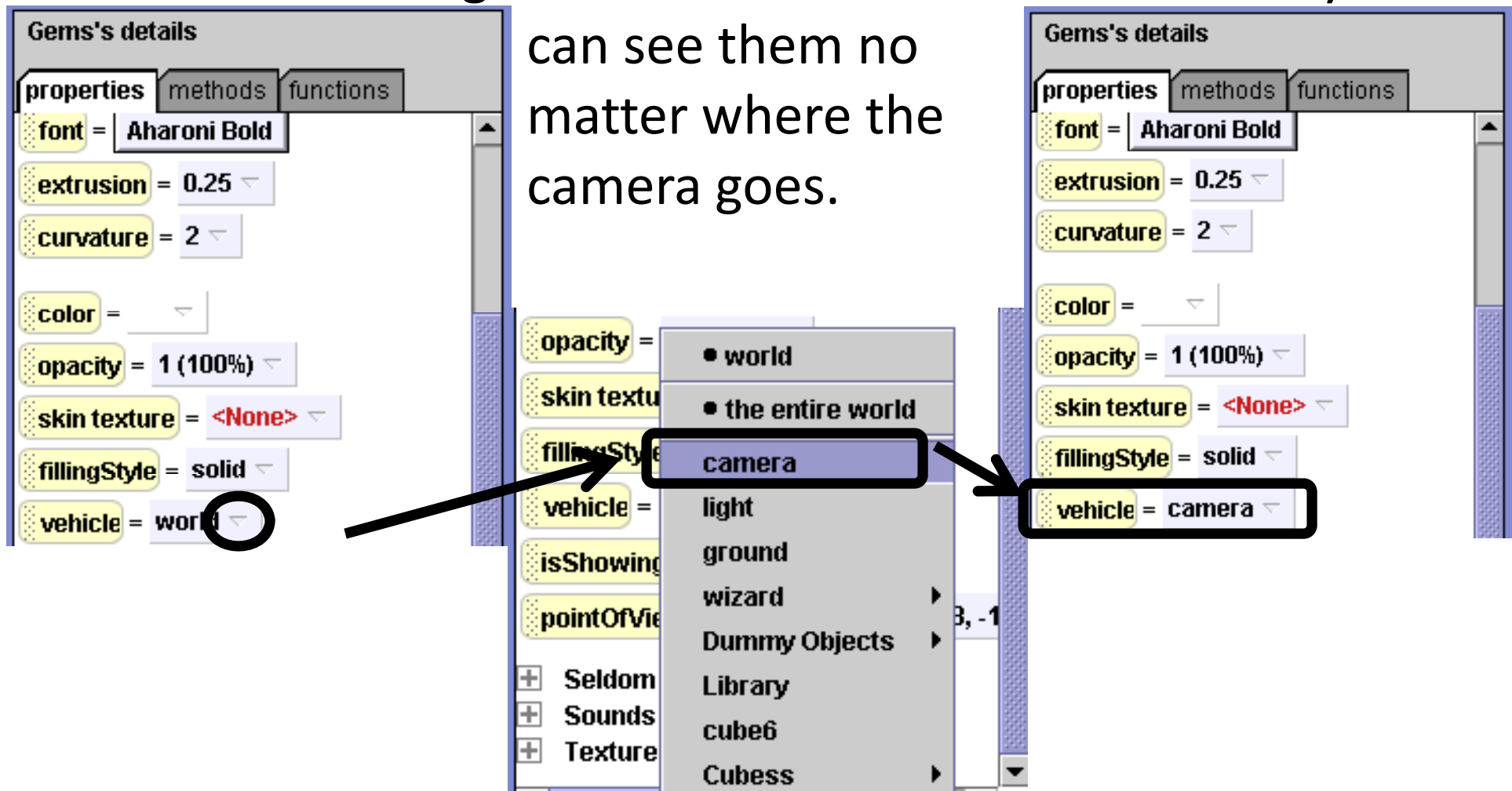


Click  
**Done**  
when  
you're  
finished.

# Setting the Text Objects

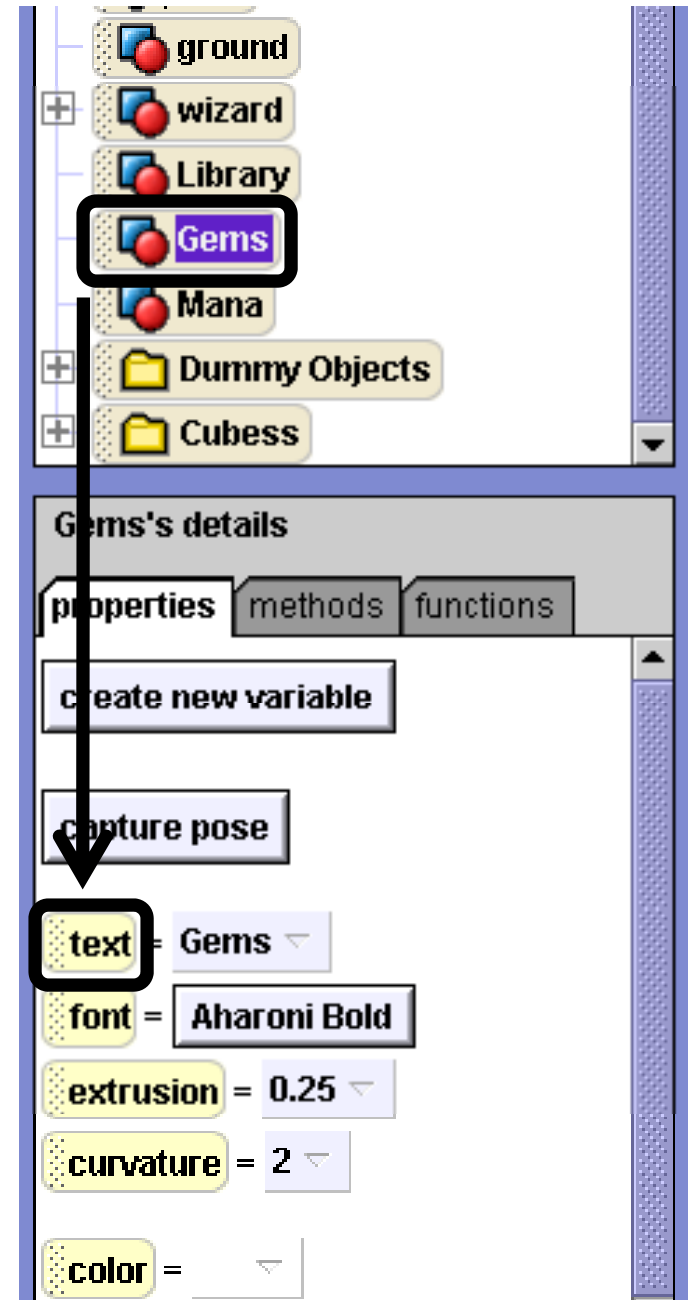
Now click on **Gems** in the object tree and go to its **properties** pane. Find the small button that says **vehicle**. Click on the down arrow next to **world**, and select **camera**. Do this same thing for **Mana**. This will make it so that you

can see them no matter where the camera goes.



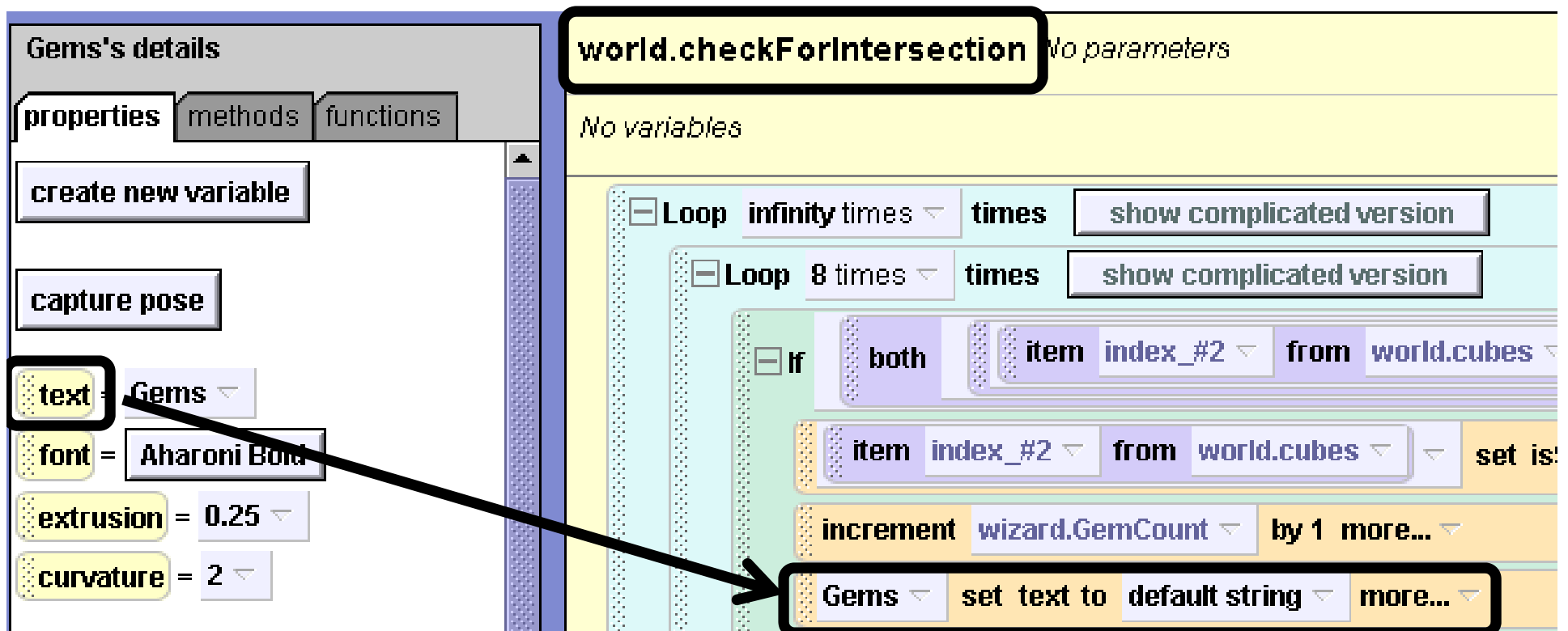
# Showing the Gem Count

Now we need to put code in our **checkForIntersection** method that connects the **GemCount** variable to the **Gems** text object. First, make sure that you are looking at **checkForIntersection** in your method editor. Then click on **Gems** in the object tree, and go to its **properties** pane. Find the small button that says **text**.



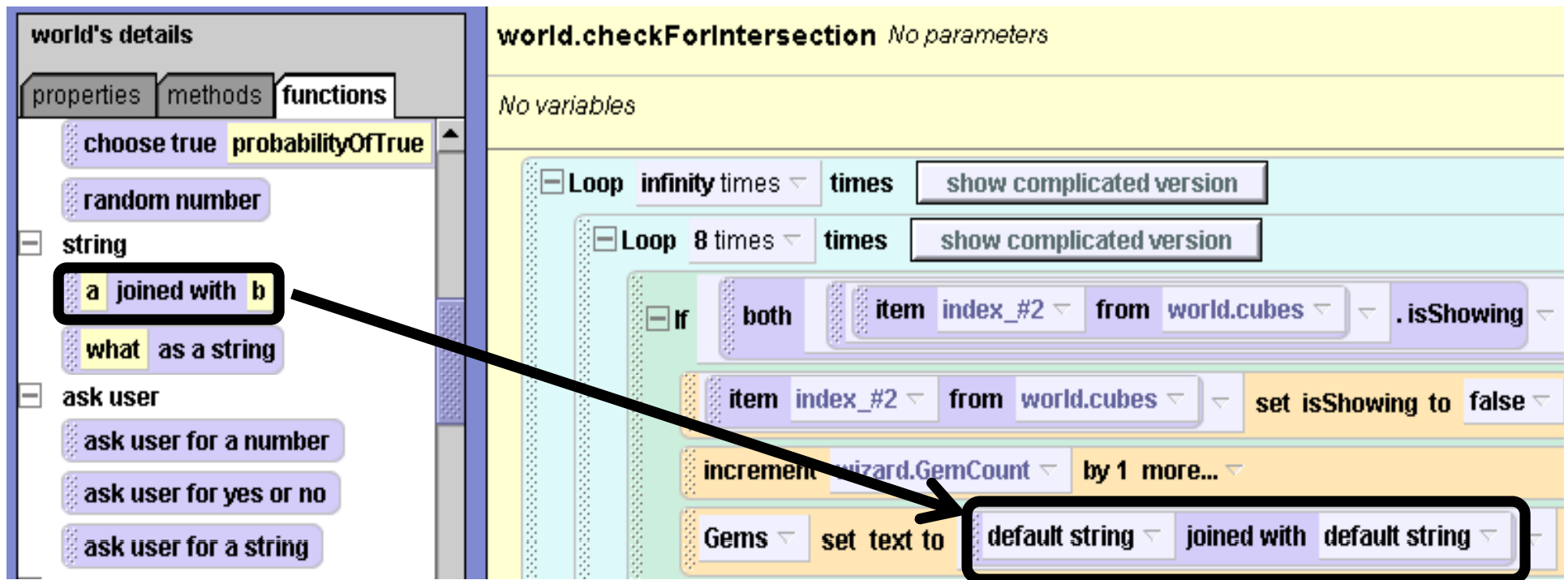
# Showing the Gem Count

Drag and drop the **text** button into the **checkForIntersection** method right under the **increment** command. Select **default string** for now.



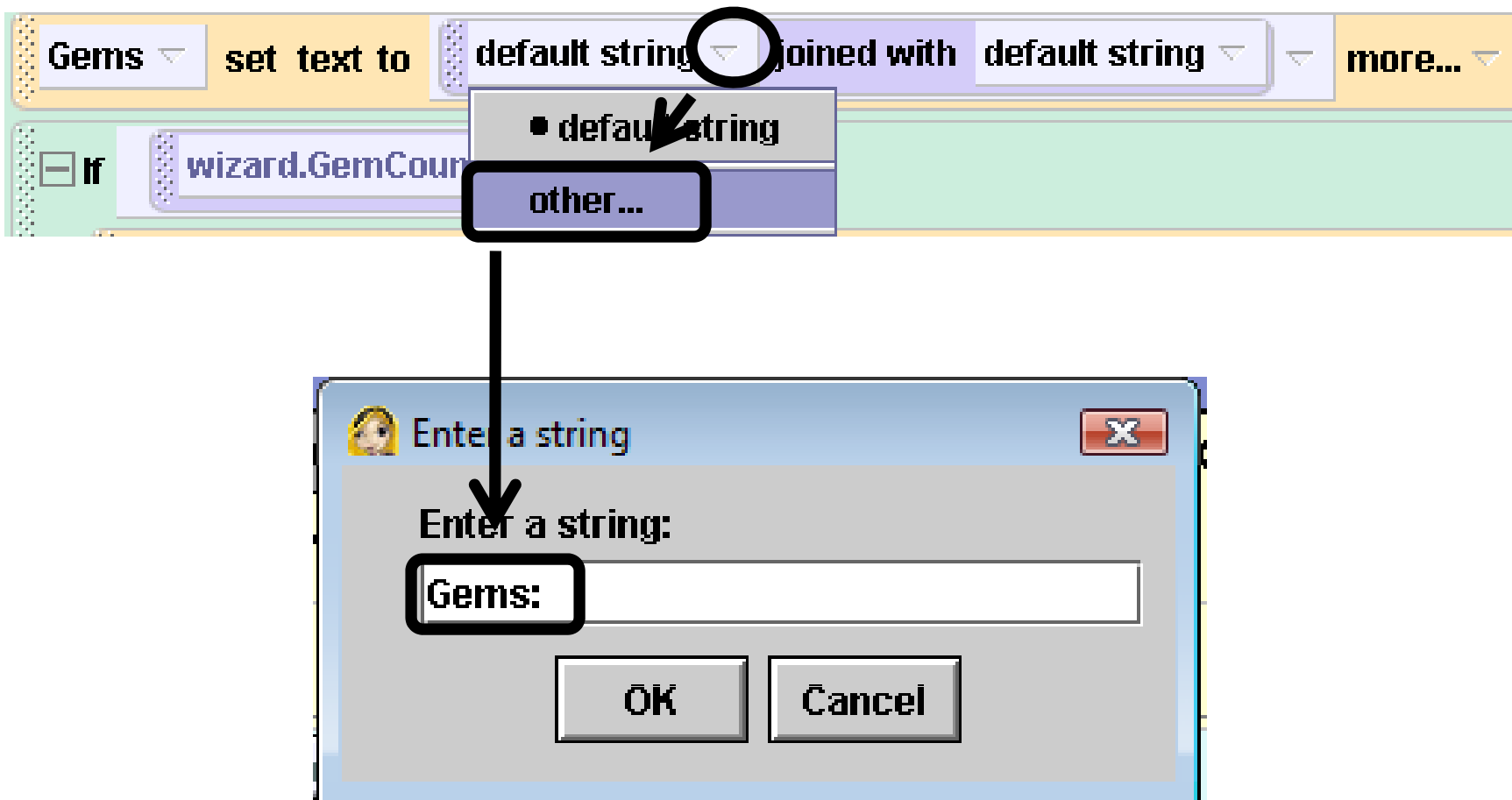
# Showing the Gem Count

Click on **world** in the object tree and go to the **functions** pane. Find **a joined with b**, and drag and drop it onto your **set text to command** where it says **default string**. Just select **default string** again.



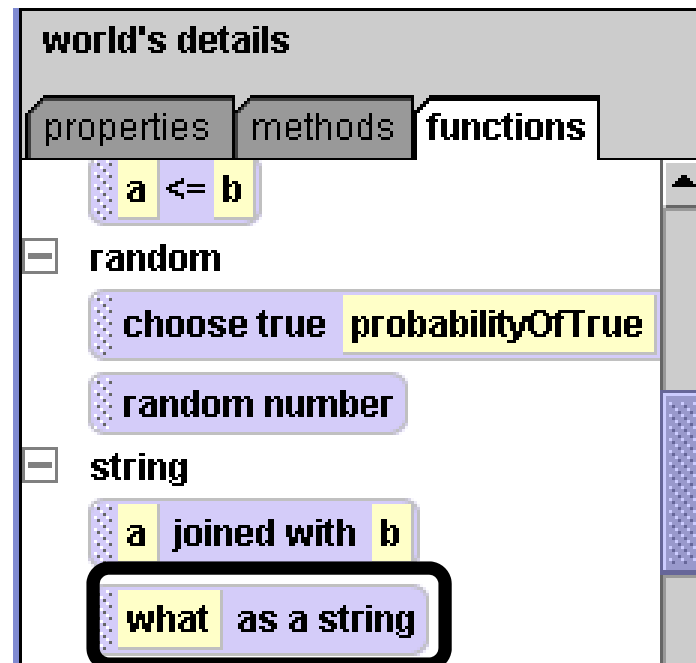
# Showing the Gem Count

Now click on the small down arrow next to the first **default string** on your **set text to** command. Select **other...** and then type in **Gems:**. Then click **OK**.



# Showing the Gem Count

Now look back at the **world's functions**. Find **what as a string** and drag and drop it on top of the second **default string** on the **set text to** command. Select **expressions**, and then **wizard.GemCount**.



# Showing the Mana

Now open up the **Teleport** method, go to **Mana**'s properties pane, and repeat all of those steps for the Mana, typing in **Mana:** for the first default string, and selecting **wizard.Mana(Magical Energy)** for **what as a string**. Your **Teleport** code will look like this:





# Showing the Mana

Now repeat those steps *again* for the **locate** method. Your code will look like this when you're done:

The image shows a Scratch code editor with the 'wizard.locate' method selected in the top toolbar. The code is as follows:

- wizard.locate** No parameters
- No variables
- If** wizard.Mana (Magical Energy) >= 30
  - wizard.Mana (Magical Energy) set value to ( wizard.Mana (Magical Energy) - 30 )
  - Mana** set text to Mana: joined with wizard.Mana (Magical Energy) as a string
  - wizard turn to face Library.closestOne ofWhich = world.cubes toWhich = wizard sizeArray = 8
- Else** Do Nothing

A black arrow points to the 'If' block, and a black box highlights the 'Mana' block.

# Testing out the Game

Now play the game. It should go through the opening as usual, except now there are two text objects on the screen also. Whenever you find a cube, your gem count should go up, and whenever you use either **Teleport** or **locate** by pressing **T** or **F**, your Mana should go down!



Now try winning the game. Can you do it?  
After you win, try winning without using Mana!

